

Foundations of R Software

Lecture 40

Strings – Display and Formatting

⋮

Substitution and Replacement of Strings

Shalabh

Department of Mathematics and Statistics

Indian Institute of Technology Kanpur

Operations with Strings: Substitution

sub and **gsub** Functions:

Within a string, we want to replace one substring with another.

Use **sub** and **gsub** to replace the first instance of a substring:

sub(old, new, string)

The **sub** function finds the first instance of the old substring within string and replaces it with the new substring.

gsub does the same thing, but it replaces all instances of the substring (a global replace), not just the first.

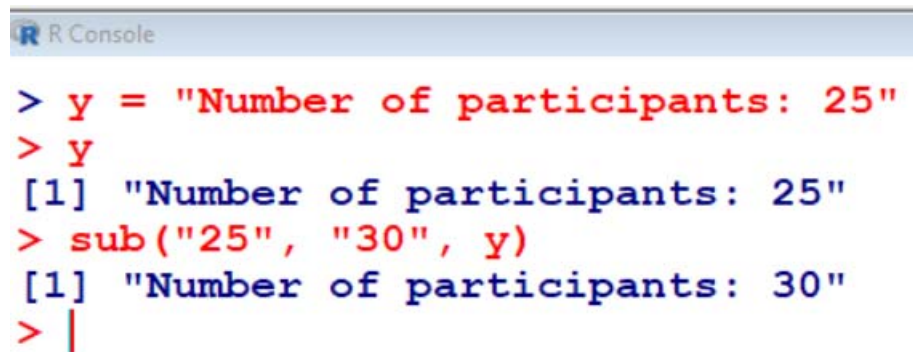
gsub(old, new, string)

Operations with Strings: Substitution

Example:

```
> y = "Number of participants: 25"
```

```
> sub("25", "30", y)  
[1] "Number of participants: 30"
```

A screenshot of an R console window showing the execution of the same code as the previous block. The console has a title bar that says "R Console". The code is entered in red text, and the output is shown in blue text. The cursor is at the end of the last line.

```
> y = "Number of participants: 25"  
> y  
[1] "Number of participants: 25"  
> sub("25", "30", y)  
[1] "Number of participants: 30"  
> |
```

Operations with Strings: Substitution

Example:

```
> y = "Mr. Singh is the smart one. Mr. Singh  
is funny, too."
```

```
> y
```

```
[1] "Mr. Singh is the smart one. Mr. Singh is  
funny, too."
```

```
> sub("Mr. Singh", "Professor Jha", y)
```

```
[1] "Professor Jha is the smart one. Mr. Singh  
is funny, too."
```

Operations with Strings: Substitution

```
R Console  
  
> y = "Mr. Singh is the smart one. Mr. Singh is funny, too."  
> y  
[1] "Mr. Singh is the smart one. Mr. Singh is funny, too."  
> sub("Mr. Singh", "Professor Jha", y)  
[1] "Professor Jha is the smart one. Mr. Singh is funny, too."  
> |
```

Operations with Strings: Substitution

Example:

```
> y = "Mr. Singh is the smart one. Mr. Singh  
is funny, too."
```

```
> gsub("Mr. Singh", "Professor Jha", y)
```

```
[1] "Professor Jha is the smart one. Professor  
Jha is funny, too."
```

Recall

```
> sub("Mr. Singh", "Professor Jha", y)
```

```
[1] "Professor Jha is the smart one. Mr. Singh  
is funny, too."
```

Operations with Strings: Substitution

R Console

```
> y = "Mr. Singh is the smart one. Mr. Singh is funny, too."
> y
[1] "Mr. Singh is the smart one. Mr. Singh is funny, too."
> gsub("Mr. Singh", "Professor Jha", y)
[1] "Professor Jha is the smart one. Professor Jha is funny, too."
>
> sub("Mr. Singh", "Professor Jha", y)
[1] "Professor Jha is the smart one. Mr. Singh is funny, too."
> |
```

Operations with Strings

R has various functions for regular expression based match and replaces.

Some functions (e.g., **grep**, **grep^l**, etc.) are used for searching for matches and functions whereas **sub** and **gsub** are used for performing replacement.

Operations with Strings: grep

The **grep** function is used for searching the matches.

(**sub** and **gsub** are used for performing replacement.)

grep : Globally search regular expression and print it

grep(pattern, x, ignore.case = FALSE) search for matches to argument **pattern** within each element of a character vector **x**.

It returns an integer vector of the indices of the elements of **x** that yielded a match.

Operations with Strings: **grep**

ignore.case : if **FALSE**, the pattern matching is case sensitive and if **TRUE**, case is ignored during matching.

value : if **FALSE**, a vector containing the (integer) indices of the matches determined by **grep** is returned, and if **TRUE**, a vector containing the matching elements themselves is returned.

Operations with Strings: grep

`grep(pattern, x, value = TRUE)` returns a character vector containing the selected elements of `x`.

```
> str = c("R Course", "exercises", "include  
examples of R language")
```

```
> grep("ex", str, value=T)
```

```
[1] "exercises"  "include examples of R  
language"
```

Operations with Strings: grep

`grep(pattern, x, value = FALSE)` returns an integer vector of the indices of the elements of `x` that yielded a match

`value = FALSE` is default.

```
> str = c("R Course", "exercises", "include  
examples of R language")
```

```
> grep("ex", str, value=F)
```

```
[1] 2 3
```

Operations with Strings: grep

```
R Console
> str = c("R Course", "exercises", "include examples of R language")
> grep("ex", str, value=T)
[1] "exercises"                "include examples of R language"
>
> str = c("R Course", "exercises", "include examples of R language")
> grep("ex", str, value=F)
[1] 2 3
> |
```

Operations with Strings: grep

`grep(pattern, x, ignore.case = TRUE)` returns a character vector containing the selected elements of `x` ignoring the case.

```
> str = c("R Course", "exercises", "include  
examples of r language", "in R software.")
```

```
> grep("R", str, ignore.case=F, value=T)
```

```
[1] "R Course"          "in R software."
```

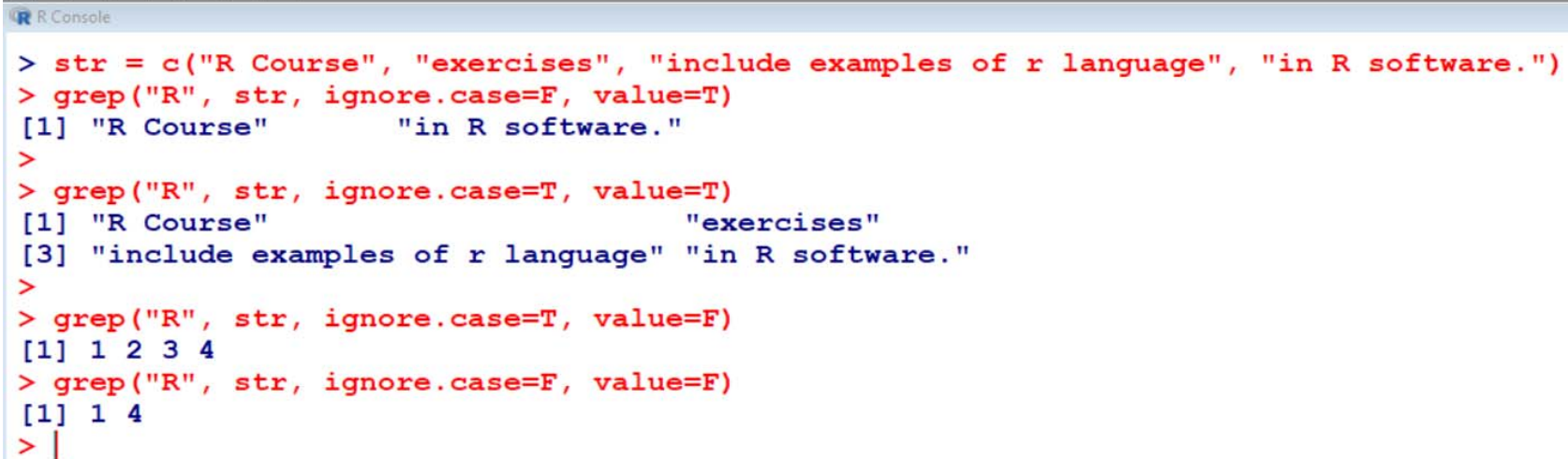
```
> grep("R", str, ignore.case=T, value=T)
```

```
[1] "R Course"          "exercises"  
[3] "include examples of r language" "in R software."
```

Operations with Strings: grep

```
> grep("R", str, ignore.case=T, value=F)  
[1] 1 2 3 4
```

```
> grep("R", str, ignore.case=F, value=F)  
[1] 1 4
```



```
R Console  
> str = c("R Course", "exercises", "include examples of r language", "in R software.")  
> grep("R", str, ignore.case=F, value=T)  
[1] "R Course"      "in R software."  
>  
> grep("R", str, ignore.case=T, value=T)  
[1] "R Course"      "exercises"  
[3] "include examples of r language" "in R software."  
>  
> grep("R", str, ignore.case=T, value=F)  
[1] 1 2 3 4  
> grep("R", str, ignore.case=F, value=F)  
[1] 1 4  
> |
```

Operations with Strings: grep

Example:

```
> x = "R course 24.07.2021"
> y = "Number of participants: 25"

> c(x,y) # Combine the two strings
[1] "R course 24.07.2021" "Number of
participants: 25"

> grep("our", c(x,y) )
[1] 1
```

"our" is in the 1st element (in the word "course"), therefore in x.
There is no "our" in y.

Operations with Strings: grep

Example:

```
x = "R course 24.07.2022"
```

```
y = "Number of participants: 50"
```

```
c(x,y) # Combine the two strings
```

```
[1] "R course 24.07.2021" "Number of  
participants: 25"
```

```
> grep("Num", c(x,y) )
```

```
[1] 2
```

“Num” is in the 2nd element (in the word “Number”), therefore in y.

There is no “Num” in x.

Operations with Strings: grep

grep function:

```
R Console
> x = "R course 24.07.2022"
> y = "Number of participants: 50"
>
> c(x,y)
[1] "R course 24.07.2022"      "Number of participants: 50"
>
> grep("Num", c(x,y) )
[1] 2
~ |
```

Operations with Strings: `grep`

`grep(pattern, x)` returns a character vector containing the selected elements of `x` and the outcome is in terms of **TRUE** and **FALSE**. Indicating if the matching is available or not.

```
> str = c("R Course", "exercises", "include  
examples of R language")
```

```
> grep("R", str)  
[1] TRUE FALSE TRUE
```

Operations with Strings: grepl

```
> str = c("R Course", "exercises", "include  
examples of R language")
```

```
> grepl("ex", str, value=T)
```

```
[1] FALSE TRUE TRUE
```

Operations with Strings: grepl

R Console

```
> str = c("R Course", "exercises", "include examples of R language")
> str
[1] "R Course"
[2] "exercises"
[3] "include examples of R language"
> grepl("R", str)
[1] TRUE FALSE TRUE
> grepl("ex", str)
[1] FALSE TRUE TRUE
> |
```